Pemrosesan Paralel Pada Model Komputasi Dokumen Ilmiah Elektronik

Setiadi Rachmat, Urip T. Setijohatmo

Jurusan Teknik Komputer dan Informatika, Politeknik Negeri Bandung, Bandung 40012
E-mail : {setiadi,urip}@jtk.polban.ac.id

ABSTRAK

Telah bermunculan *search engine* baru atau optimasinya untuk mengantisipasi pertumbuhan jumlah dokumen pada Web memungkinkan setiap orang untuk memperoleh informasi dalam bentuk dokumen elektronik lebih banyak. Untuk suatu koleksi hasil pengumpulan dokumen berupa perpustakaan artikel ilmiah elektronik yang terdistribusi dibutuhkan *search engine* di lingkungan intranet. Namun masih minimnya *search engine* di lingkungan intranet menjadi kendala dimana arsitekturnya dan kebutuhannya berbeda dengan Web, sehingga memanfaatkan *search engine* untuk Web tidak efektif bila diimplementasikan di lingkungan intranet. Pada perkembangan lainnya, teknologi perangkat keras sudah mencapai kemampuan sebuah desktop dengan *multicore*. Dengan *computing power* yang semakin besar maka kebutuhan komputasi yang besar dapat dilakukan secara paralel memanfaatkan *multicore* tersebut. Penelitian ini merupakan pengembangan *search engine* di lingkungan intranet, khususnya meningkatkan kinerja dari perangkat lunak menjadi Sistem Layanan Dokumen yang berkemampuan pemrosesan secara paralel menggunakan server *multicore* rakitan. Walaupun pada penelitian ini belum secara penuh mendukung pemrosesan paralel namun merupakan langkah awal dimana pemrosesan paralel dilakukan terhadap model pemrosesan yang mewakili prinsip komputasi sub proses yang berpotensi *bottleneck* yang memperlambat kinerja proses. Penelitian ini telah menghasilkan suatu arsitektur perangkat keras server rakitan yang berkemampuan pemrosesan paralel menggunakan *middleware* MPI (*Message Passing Interface*) dengan model komputasi paralel SPMD (*Single Program Multiple Data*). Telah pula teridentifikasi *bottleneck* dan potensi pengembangan secara paralel dan diputuskan pemodelan berupa perhitungan perkalian matrix. Percobaan telah dilakukan pada server hasil rakitan untuk menguji apakah hasil pembangunan benar dengan kasus menghitung Phi. Adapun algoritma perkalian matrix paralel yang digunakan adalah *shift-and-compute* dengan asumsi n berukuran *perfect square*.

Kata Kunci
Pemrosesan paralel, SPMD, message passing, MPI, multicore

1. PENDAHULUAN

Keberadaan World Wide Web merepresentasikan era informasi. Menurut [2] lautan informasi ini mengandung 2.3 milyar dokumen digital dan para analis memprediksi jumlah dokumen pada Web akan berkembang delapan kali pada tahun 2000 - dan 100 kali pada dekade berikutnya. Hal tersebut mendapat tanggapan dari peneliti Teknologi Informasi (IT) dengan melakukan banyak penelitian yang bertujuan untuk mengoptimalisasi *search engine* agar dapat memperoleh suatu informasi yang tepat dan sesuai dengan kebutuhan seseorang dari jumlah dokumen yang banyak. Telah bermunculan *search engine* baru atau optimasinya untuk mengantisipasi pertumbuhan jumlah dokumen pada Web [3]. Kombinasi dari dua hal tersebut memungkinkan setiap orang untuk memperoleh informasi dalam bentuk dokumen elektronik sebanyak-banyaknya. Masalahnya adalah semakin banyaknya dokumen yang terkumpul (disimpan pada *offline/standalone* komputer atau intranet) bukan berarti semakin mudah orang tersebut me-*retrieve*/mendapatkan kembali dokumen yang dibutuhkan, tetapi justru akan berdampak pada sulitnya pencarian informasi yang spesifik pada suatu dokumen. Hal ini disebabkan oleh masih minimnya *search engine* di lingkungan *offline/standalone* dan intranet, dimana arsitekturnya dan kebutuhannya berbeda dengan Web [4]. Sehingga memanfaatkan *search engine* untuk Web tidak efektif bila diimplementasikan untuk lingkungan *offline/standalone* dan intranet.

Pada perkembangan lainnya, teknologi perangkat keras sudah mencapai kemampuan sebuah *desktop* dengan *multicore*. Dengan kemampuan komputasi yang semakin besar maka kebutuhan komputasi yang besar dapat dilakukan secara paralel memanfaatkan *multicore* tersebut. Kendala yang muncul adalah masih mahalnya harga komputer dengan spesifikasi ini. Kendala ini dapat ditangani dengan merakit sendiri komputer server ber-arsitektur beberapa prosesor serial tunggal berkemampuan pemrosesan paralel.

Atas dasar hal-hal tersebut, dapat dikembangkan suatu mekanisme layanan dokumen dalam lingkungan intranet yang berkemampuan pemrosesan paralel terdistribusi menggunakan perangkat keras server komputasi rakitan.

2. TINJAUAN PUSTAKA

Pada bagian ini akan dijabarkan pustaka-pustaka yang relevan dengan penyelesaian masalah penelitian yang dikelompokkan sebagai berikut:

- Kebutuhan *Search Engine*

  Seperti yang dijelaskan pada [2], dunia mengalami pertumbuhan eksponensial informasi digital. Pertumbuhan informasi ini didorong oleh percepatan adopsi teknologi digital. Tumpukan statistik dari berbagai sumber berbagi tema umum yaitu tingkat pertumbuhan untuk informasi digital tak terbantahkan dan sangat tinggi. Pada tahun 1995, lebih dari 90% dari dokumen itu dalam bentuk kertas. Selama beberapa tahun terakhir, bagaimanapun, telah terjadi penurunan signifikan dalam jumlah dokumen kertas saja. Pada tahun 2005, para analis memprediksi bahwa hanya 30% dari dokumen akan tetap di atas kertas. Tren yang jelas adalah bahwa hampir semua informasi dalam dunia bisnis saat ini dimanipulasi, dimodifikasi, diproduksi dan pindah dalam bentuk digital.

  Sementara itu suatu statistik lain yang dipaparkan pada [8] menyatakan bahwa sebuah studi yang dirilis baru-baru ini mengkuantifikasi seberapa cepat alam semesta digital berkembang.

  Informasi dunia dua kali lipat setiap dua tahun, dengan menakjubkan 1,8 zettabytes diciptakan dan direplikasi pada tahun 2011, menurut IDC. Jumlah tersebut adalah menyatakan informasi yang besar - sama dengan 1,8 triliun gigabyte tersimpan dalam 500 *file* kuadriliun. Lautan informasi seperti itu membutuhkan dukungan *search engine*. Seperti yang dijelaskan pada [3] bermunculan *search engine* baru atau optimasinya untuk mengantisipasi pertumbuhan jumlah dokumen pada Web. Bahkan bermunculan organisasi yang membantu pertumbuhan pemasaran mesin pencari.

- Pengembangan *Search Engine* yang ada di lingkungan Intranet [4]

  Dengan fasilitas internet, setiap orang biasanya mendapatkan dokumen yang dicarinya dengan cara *googling* dan mengunduh yang kemudian dokumen sebagai hasilnya disimpan masing-masing.

  Masalahnya adalah semakin banyaknya dokumen yang terkumpul (disimpan pada *offline/standalone* komputer atau intranet) bukan berarti semakin mudah orang tersebut me-*retrieve*/mendapatkan kembali dokumen yang dibutuhkan, tetapi justru akan berdampak pada sulitnya pencarian informasi yang spesifik pada suatu dokumen. Hal ini disebabkan oleh masih minimnya *search engine* di lingkungan *offline/standalone* dan intranet, dimana arsitekturnya dan kebutuhannya berbeda dengan Web [9]. Sehingga memanfaatkan *search engine* untuk Web tidak efektif bila diimplementasikan untuk lingkungan *offline/standalone* dan intranet. Sedangkan pada [4] membandingkan kinerja Google dengan layanan basis data perpustakaan.

- Perangkat Lunak Sistem Manajemen Dokumen yang ada

- Perangkat lunak System Manajemen Dokumen yang ada [1] telah mencakup semua proses mulai dari Akuisisi, Ekstraksi, Penyimpanan dan *Retrieval*, namun dirasakan kinerjanya masih terbatas sehingga perlu dikembangkan suatu system yang mampu memaksimalkan resources yang dipunyai, multi pemroses dan terdistribusi. Untuk kebutuhan penelitian yang sedang dilakukan, perlu dianalisis lebih detil perilaku semua sub proses sehingga selain akan ditemukan hal-hal yang memperlambat kinerja, juga pemilihan model proses paralel dalam memanfaatkan semua resources

- Tools *Message Passing Interface* MPI.

  Semua program MPI harus mengikuti struktur umum seperti berikut:

  - *Include* MPI *header file*
  - Deklarasi variabel
  - Inisialisasi lingkungan MPI
  - Melakukan komputasi dan pemanggilan komunikasi MPI
  - Menutup komunikasi MPI

3. HASIL DAN PEMBAHASAN

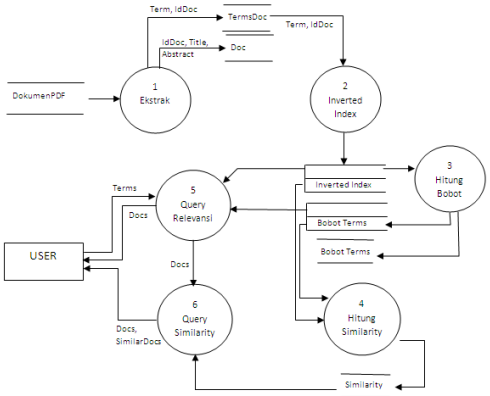
3.1 Analisis Kebutuhan Secara Umum

Dokumen yang menjadi perhatian merupakan dokumen penelitian berformat pdf. Dengan fasilitas internet, dosen setiap jurusan mendapatkan dokumen yang dicarinya dengan cara *googling* dan mengunduh yang kemudian dokumen sebagai hasilnya disimpan masing-masing. Pembaca dokumen biasanya mengamati abstrak untuk memastikan isinya sesuai atau tidak dengan yang dicarinya. Bila sesuai dokumen dibaca, dan bila belum puas pembaca ini akan mencari dokumen yang mirip. Intensitas yang tinggi mengakibatkan penumpukan dokumen yang tidak efisien, pengaksesan kembali yang sulit, dan sharing yang belum dimungkinkan. Dari uraian di atas beberapa hal *requirement* adalah sebagai berikut:

- Ekstraksi dokumen penelitian. Ekstraksi adalah mendapatkan kata/*term* yang terkandung dalam dokumen. Selain memperhatikan aturan pembentukan kata, juga section sebagai pemisah isi sebagai aturan format penelitian.
- Penyediaan mekanisme penyimpanan data yang menghindari duplikasi, kemudahan akses dan

memungkinkan sharing tanpa harus mengetahui posisi fisik dokumen.

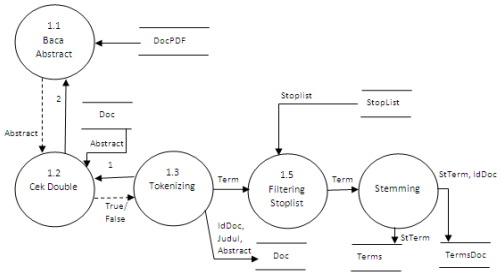
- Penyediaan kemampuan (fitur) menentukan relevansi *terms* pada dokumen dokumen.
- Penyediaan kemampuan menentukan kemiripan dokumen.
- Penyediaan kemampuan *query* mendapatkan dokumen sesuai istilah (*terms*) yang diinginkan.
- Penyediaan kemampuan *query* mendapatkan dokumen yang mirip dengan dokumen tertentu.



Gambar 1: Kebutuhan Sistem Pemrosesan Dokumen

3.2 Identifikasi *Bottleneck* dan Potensi Pemrosesan Paralel

Proses yang terlibat adalah ekstraksi dari dokumen pdf menjadi *term-term*, lalu proses membentuk *inverted index*, dilanjutkan dengan proses menghitung bobot *term* dan similaritas dokumen dan akhirnya adalah pemrosesan *query* relevan dengan *term query* dan *query* dokumen yang mirip.



Gambar 2: Subsistem Ekstraksi Dokumen

Proses Ekstraksi Dokumen

Masalah ekstraksi terdiri dari beberapa subproses, yaitu: tokenizing, verifikasi stopword dan stemming. Tujuan dilakukannya *parsing* atau tepatnya *tokenizing* ini adalah untuk mendapatkan *term-term* yang nantinya akan diindeks. Sub proses Membaca Abstrak memproses sekitar 100 dokumen untuk masing-masing unit. Seperti diketahui dalam mekanisme layanan dokumen setiap unit dipasang satu server, sehingga setiap hari masing-masing unit memproses hanya sekitar 100 yang diproses secara bersamaan. Untuk cek *double* masing-masing dari 100 dokumen tersebut akan dicari padanannya pada basis data berukuran ratusan ribu dokumen dimana bila ditemukan berarti terdeteksi duplikasi (dokumen sudah ada). Proses ini bila hanya menggunakan cek *double* membandingkan abstrak saja hanya dapat dijalankan secara sequential satu persatu setiap dokmen yang di-load dan dibandingkan dengan ratusan ribu dokumen pada basis data. Maka sub proses ini teridentifikasi merupakan *bottleneck*. Walaupun untuk pengembangan yang sebenarnya subproses ini berpotensi diperlakukan secara paralel dengan prinsip SPMD (*single program multiple data*) dimana data dibagi menjadi beberapa kumpulan yang setiap kumpulan diassign subproses ini pada *core* tertentu menggunakan multiproses), Dengan penambahan mekanisme cek *double* yang dikembangkan sedemikian rupa (misal dari *file size* dan penggunaan *multithread* pada *multicore*) secara signifikan akan menaikkan kinerja system layanan dokumen.

Pada setiap server sub proses *tokenizing*, *stemming*, dan *filtering* masing-masing memproses sekitar 100 dokumen yang masing-masing mengandung rata-rata 7000-an *term* sehari. Subproses *tokenizing*, *stemming*, dan *filtering* dapat diperlakukan menggunakan *multithread* sehingga bukan merupakan *bottleneck* (walaupun bisa dikembangkan dengan memperlakukan sub proses ini secara paralel) sehingga bukan termasuk kajian penelitian ini.
Sedangkan pada proses pembentukan *Inverted Index*, ketika suatu *term* hasil *tokenizing* ingin didaftarkan pada *inverted index* akan dicek terlebih dahulu sudah adakah di daftar *term*nya. Kalau belum, daftarkan dan menetapkan jumlah=1, bila sudah, jumlah *term* pada dokumen yang dimaksud bertambah satu. Seperti diketahui, jumlah *term* dalam suatu dokumen adalah banyak, dengan jumlah dokumen yang bertambah maka jumlah *term* menjadi semakin banyak pula. Untuk mempercepat proses tersebut di atas dibutuhkan penstrukturan *inverted index* menggunakan mekanisme pengindeksan. Dan untuk keperluan ini digunakan *binary search tree* (BST).

Subproses ini dilakukan di server komputasi dan merupakan langkah persiapan untuk menghitung bobot dengan metode TF/IDF(*term frequency–inverse document frequency*) dan similaritas dokumen dengan metode LSA(*Latent Semantic Analysis*). Server komputasi menerima kiriman dari setiap server lain dan mengakumulasinya pada matrik *term*-dokumen yang berstruktur *inverted index*. Subproses ini setiap mengakumulasi *worstcase* adalah mendaftarkan (10) x(100 x 7000) *term* kedalam *inverted index*, dan ukuran tersebut cukup besar sehingga berpotensi menjadi *bottleneck*. Oleh karena itu sub proses ini diperlakukan sebagai pemrosesan paralel dan termasuk kajian penelitian.
Proses Pembobotan *Term* dibutuhkan dalam menentukan peringkat dokumen (*document ranking*) – pada operasi pembobotan dokumen, dimana hal itu dilakukan untuk mencari besarnya relevansi antara dokumen dengan *query*.

Metode pembobotan *term* yang digunakan pada[4] adalah TF-IDF. Bobot TF–IDF adalah suatu bobot yang sering digunakan dalam *information retrieval* dan *text mining*. Adapun rumus umum dari TF-IDF adalah sebagai berikut :

$$W_{ij} = tf_{ij} \times \left(\left(\log\frac{N}{n}\right) + 1\right) \quad (1)$$

dimana

W_{ij} = bobot kata *term* t_j terhadap dokumen d_i

tf_{ij} = jumlah kemunculan kata / *term* t_j dalam d_i

N = jumlah semua dokumen yang ada

n = jumlah dokumen yang mengandung kata / *term* t_j

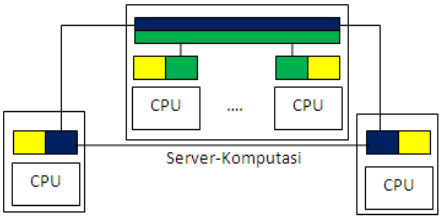
Dari ilustrasi diperoleh kesimpulan bahwa sub proses ini memerlukan dukungan mekanisme pemrosan paralel karena akan memproses sel sejumlah sangat besar sel bobot. Oleh karena itu sub proses hitung bobot dengan metode TF/IDF berpotensi *bottleneck* dan memerlukan dukungan mekanisme pemrosesan paralel. Lebih jauh menurut [8], terdapat beberapa metode perhitungan similaritas semantik dokumen. Salah satu dari metode yang ditinjau adalah LSA. LSA melakukan singular value decomposition (SVD) terhadap matrik yang sudah dibentuk. Matrik yang direpresentasikan menggunakan SVD akan diuraikan menjadi 3 (tiga) komponen matrik, yaitu matrik vektor singular kiri, martik nilai singular, dan matrik vektor singular kanan atau dapat dirumuskan sebagai berikut:

$$A_{mn} = U_{mm} \cdot S_{mn} \cdot V_{nn}^{T} \quad (2)$$

Salah satu keperluan penggunaan SVD yaitu dibutuhkan pada pencarian nilai *similarity* untuk *term* dan dokumen yang dilakukan melalui perkalian matrik *U*, *S* dan V^T hasil reduksi. Vektor baris digunakan untuk mencari nilai *similarity term* dan vektor kolom untuk *similarity* dokumen. Untuk pencarian dokumen *similarity* mengalikan matrik *V* dengan *S* (*V*·*S*) dan perhitungan vektor barisnya sebagai nilai *similarity*nya. Sama seperti yang diilustrasikan pada subproses pembobotan dengan TF/IDF, sub proses menghitung similaritas dokumen ini berpotensi *bottleneck* dan memerlukan dukungan mekanisme pemrosesan paralel.
Pada pemrosesan *query*, dua subproses yaitu pemrosesan *query* relevansi dokumen mendapatkan dokumen yang relevan dan *query* similaritas dokumen telah dijalankan secara terdistribusi dan walaupun dapat dikembangkan dengan dukungan mekanisme pemrosesan *query*, namun bukan termasuk kajian penelitian ini karena bukan merupakan *bottleneck* dimana mekanisme akses sudah memanfaatkan dukungan *indexing* pada basis data dan pemrosesan terdistribusi.

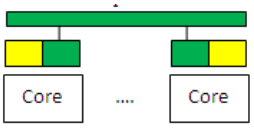
3.3 Pemodelan dan Spesifikasi Perilaku Komputasi Paralel

Seperti yang telah dideskripsikan pada bagian sebelumnya potensi dukungan mekanisme komputasi paralel diputuskan dilakukan pada subproses pembentukan *inverted index*, subproses menghitung bobot dengan metode TF/IDF dan menghitung similaritas dokumen. Pemodelan yang diputuskan adalah paralelisasi perkalian matrik dalam rangka perhitungan SVD. Adapun arsitektur umum desain sistem akan terdiri dari beberapa server yang masing-masing mempunyai beberapa tugas berbeda (*distributed specific jobs*) dengan komunikasi antar proses menggunakan *middleware* ICE dan MPI.



Gambar 3: Arsitektur Umum

Arsitektur khusus yang menjadi kajian penelitian ini tidak termasuk ICE di dalamnya. Secara pemrosesan non paralel, ICE sudah digunakan jadi tidak akan dibahas lebih lanjut. Dengan perkataan lain fokus penelitian ada pada Server komputasi dimana kajian arsitektur perangkat lunak hanya memperhatikan *middleware* MPI saja. Perangkat keras dikomposisi dari 16 *Core* yang terhubung dengan bus. Masing-masing CPU mempunyai memori lokal.



Server Komputasi

Gambar 4: Arsitektur Kajian

Gambar 5: Server Rakitan

Algoritma Paralel Perkalian Matrik

nloop=sqrt(n)
//Initial alignments
Baris ke i s/d n digeser kekiri sebanyak baris ke
Kolom ke i s/d n digeser keatas sebanyak kolom ke

```
//Lakukan perkalian matrix
for(int i=0; i<nloop; i++) do begin
   for(int j=0; j<nloop; j++) do begin
        C2[i][j]=new Matrx(nloop);
        C2[i][j].isi=PerkalianMatrix(A[i][j].isi,B[i][j].isi,nloop);
        C[i][j].isi=PertambahanMatrix(C[i][j].isi,C2[i][j].isi, nloop);
   endfor
endfor
// Alignment penggeseran selanjutnya dan perkalian matrik dilakukan
sebanyak sqrt(n)-1
for(int lup=1; lup<nloop; lup++) do begin
   Setiap Baris Matrix A semua kolom digeser kekiri
   Setiap Kolom Matrix B semua baris digeser keatas
   // Lakukan perkalian matrix
   for(int i=0; i<nloop; i++) do begin
        for(int j=0; j<nloop; j++) do begin
             C2[i][j]=new Matrx(nloop);
             C2[i][j].isi=PerkalianMatrix(A[i][j].isi,B[i][j].isi,nloop);
             C[i][j].isi=PertambahanMatrix(C[i][j].isi,C2[i][j].isi,
               nloop);
        endfor
   endfor
endfor
```

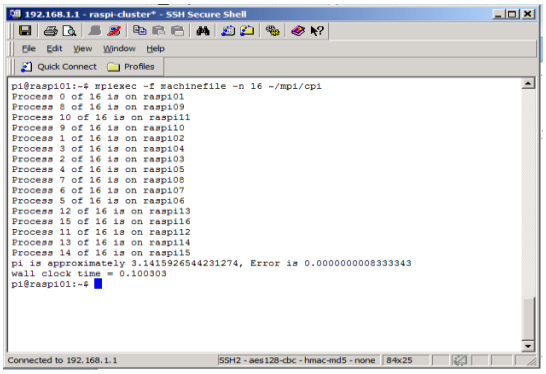
3.4 Percobaan

Seperti yang telah dijelaskan pada bagian sebelumnya, model komputasi yang mewakili Sistem Layanan Dokumen adalah pemrosesan matrik. Di sisi lain arsitektur komputer rakitan paralel memiliki secara instrinsik kemampuan bermodel SPMD. Oleh karena itu rancangan pengujian yang akan dicobakan adalah berkenaan dengan operasi pada matrik secara SPMD.

Rancangan Percobaan dan Hasil Percobaan

Pada perhitungan SVD terdapat perkalian suatu matrik *A* dengan transposenya A^T dan antara suatu matrik *A* dengan suatu matrik lain B. Sesuai dengan perkiraan jumlah rata-rata pertambahan dokumen unik per unit setiap hari 10, jumlah unit=5, dengan jumlah *term* unik masing-masing dokumen=500, maka total pertambahan dokumen 50 dan jumlah pertambahan *term* 25.000. Dari sisi perangkat keras perlu diketahui sampai seberapa signifikan pertambahan jumlah CPU mempengaruhi kinerja komputasi.

Untuk pengujian mesin server rakitan apakah dapat berjalan normal, dilakukan dengan menggunakan perhitungan phi pada 16 *processor element* (PE).



Gambar 6: Hasil Percobaan perhitungan phi dengan 16 PE

Dari hasil percobaan diketahui bahwa mesin server rakitan berjalan sebagaimana yang direncanakan.

4. KESIMPULAN

Penelitian ini merupakan perbaikan kinerja waktu pemrosesan dokumen dari penelitian sebelumnya [1] pada server komputasi dari server tunggal menjadi *multicore* berkemampuan pemrosesan paralel. Telah diimplementasi perhitungan phi secara paralel dan perkalian matrix berdimensi *perfect square* sebagai percobaan. Permasalahan menghitung phi secara paralel adalah standar yang biasa dilakukan sedangkan permasalahan perkalian matrik dipilih untuk mewakili model komputasi dari perhitungan pada server komputasi[1]. Pencapaian penelitian saat ini adalah telah dilakukannya penyusunan algoritma komputasi paralel, berhasil merakit dan memverifikasi mesin paralel yang akan digunakan sebagai server komputasi. Target selanjutnya adalah implementasi algoritma komputasi paralel pada server komputasi.

UCAPAN TERIMA KASIH

Sebagai ucapan terima kasih kami tujukan kepada pihak UPPM yang telah menjembatani terlaksananya penelitian ini, juga rekan dosen atas atmosfir penelitian yang menginspirasi.

DAFTAR PUSTAKA

[1] Urip T Setijohatmo, Setiadi Rachmat, Irwan Setiawan , *Sistem Manajemen Dokumen: Pengorganisasian dan Temu Kembali Dokumen Artikel Ilmiah Elektronik*, Proceeding Industrial Research Workshop dan Seminar Nasional Sains Terapan (IRWNS) 2010, Politeknik Negeri Bandung, Nopember 2010.

[2] [http://www.tdan.com/ view-articles/4917, 23 Maret 2013.

[3] http://www.articlesbase.com/business- articles/ sempo-search-engine-marketing-301029.html, 23 Maret 2013.

[4] Jan Brophy, David Bawden, (2005) "Is Google enough? Comparison of an internet search engine with academic

library resources", Aslib Proceedings, Vol. 57 Iss: 6, pp.498 - 512
[5] Brandon Pincombe, Comparison of human and latent semantic analysis (LSA) judgments of pairwise document similarities for a news corpus, Defence Science and Technology Organisation Research Report DSTO–RR–0278, 2004.
[6] Lee M. D., Pincombe B., & Welsh M., An empirical evaluation of models of text document similarity, 27th Annual Meeting of the Cognitive Science Society, 2005
[7] Thomas Hofmann, Information Retrieval- Retrieval Models, Lecture 5 – October 24th, 2007.
[8] http://www.infodocket.com/2011/06/28/statistics-daily-deluge-of-digital-data-expected-to-get-even-worse/, 2 September 2013.
[9] http://www.nngroup.com/articles/the-difference-between-intranet-and-internet-design/, 23 Maret 2013.